AISSMS

INSTITUTE OF INFORMATION TECHNOLOGY

ADDING VALUE TO ENGINEERING

An Autonomous Institute Affiliated to Savitribai Phule Pune University

Approved by AICTE, New Delhi and Recognised by Govt. of Maharashtra

Accredited by NAAC with 'A+' Grade

A

Lab Practice IV Mini Project REPORT

ON

# FOREST FIRE DETECTION USING SATELLITE IMAGERY

BY


**CHINMAY BELDAR <72289362M>**

**SAGAR JADHAV <72145604K>**

**PRATHAM PAWALE<72145795K>**

**UPENDRA TARAL<72289369J >**

UNDER THE GUIDANCE OF

**Prof. Mrunal K. Pathak**

DEPARTMENT OF INFORMATION TECHNOLOGY

ALL INDIA SHRI SHIVAJI MEMORIAL SOCIETY'S

INSTITUTE OF INFORMATION TECHNOLOGY

PUNE– 411001

SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE


**2023-24**

AISSMS
INSTITUTE OF INFORMATION TECHNOLOGY
ADDING VALUE TO ENGINEERING
An Autonomous Institute Affiliated to Savitribai Phule Pune University
Approved by AICTE, New Delhi and Recognised by Govt. of Maharashtra
Accredited by NAAC with 'A+' Grade

# CERTIFICATE

This is to certify Lab Practice- IV Mini Project Report entitled "**Forest Fire Detection Using Satellite Imagery**" being submitted by

Chinmay Beldar (72289362M)

Sagar Jadhav (72145604K)

Pratham Pawale (72145795K)

Upendra Taral (72289369J)

Is a record bonfied work carried out by group under the supervision and guidance of Dr. Mrunal Pathak in partial fulfillment of the requirement for BE (Information Technology Engineering) _ 2019 course of Savitribai Phule University, Pune in the academic year 2023-2024

Date : 26-10-23

Place : Pune

| Name of the mentor | Head of the department |
|---|---|
| Dr. Mrunal Pathak | Dr. Meenakshi A. Thalor |

AISSMS
INSTITUTE OF INFORMATION TECHNOLOGY
ADDING VALUE TO ENGINEERING
An Autonomous Institute Affiliated to Savitribai Phule Pune University
Approved by AICTE, New Delhi and Recognised by Govt. of Maharashtra
Accredited by NAAC with 'A+' Grade



# Table of Contents

AISSMS

INSTITUTE OF INFORMATION TECHNOLOGY

ADDING VALUE TO ENGINEERING

An Autonomous Institute Affiliated to Savitribai Phule Pune University

Approved by AICTE, New Delhi and Recognised by Govt. of Maharashtra

Accredited by NAAC with 'A+' Grade


# Acknowledgement


We convey our warm gratitude and sincere thanks towards Dr Mrunal K. Pathak Ma'am for her valuable guidance, constant encouragement, and valuable suggestions, without which the present project would not have come to its shape. We are equally grateful to all present faculty members of Information Technology department for helping in choosing right tech stack as well as clearing doubts from time to time in accomplishing our project work. We express our gratitude to the Author, editors, and publishers of the books which helped us in completing preparatory research. We would also thank to all of them who gave us valuable inputs which led to completion and improvement in the project.

1. **Chinmay Beldar**
2. **Sagar Jadhav**
3. **Pratham Pawale**
4. **Upendra Taral**

**Title**: Forest Fire Detection Using Satellite Imagery.

**Aim**: The aim of this project is to develop a real-time wildfire detection system using satellite imagery. This system will be able to detect wildfires early, so that firefighters can respond quickly and minimize damage.

**Objective**: The main objectives of this project are to:

- Develop a deep learning model capable of identifying wildfire-affected regions in satellite imagery.
- Utilize the model to estimate the extent of burned areas and $CO_2$ emissions caused by wildfires.
- Contribute to environmental conservation and disaster management by providing accurate and timely information to relevant authorities and organizations.

**Scope**: The scope of this project encompasses the following areas:

- Utilizing deep learning techniques to analyze satellite imagery for wildfire detection.
- Developing an algorithm to estimate the size and impact of wildfires, including $CO_2$ emissions, and burned areas.
- The project focuses on the global application of the system to aid in early wildfire detection, with potential for adaptation to specific regions.

**Theory**:

The project is based on the premise that the combination of real-time satellite imagery and deep learning can significantly improve early wildfire detection. The deep learning model is designed to recognize patterns and anomalies in the imagery that are indicative of active wildfires, thereby providing an early warning system. Additionally, the estimation of $CO_2$ emissions and burned areas contributes to a better understanding of the environmental impact of wildfires.

Wildfires pose a significant threat to the environment and human lives. Early detection is crucial to mitigate their impact. This project utilizes deep learning techniques and satellite imagery to detect wildfires. The U-Net deep learning model is employed for image segmentation, allowing the system to identify burning areas with high accuracy. The system also includes a CO2 emission calculator to estimate the environmental impact of the wildfire.

# INTRODUCTION

Wildfires, a global menace, pose a substantial threat to both the environment and human well-being. They have the capacity to inflict extensive harm on property, infrastructure, and human life, emphasizing the critical need for early detection. Early detection of wildfires is pivotal in minimizing the catastrophic impact these events can have. Traditionally, methods of wildfire detection have been limited by factors such as monitoring delays and restricted coverage. However, modern technology and innovation have paved the way for more effective solutions. This project capitalizes on the power of satellite imagery and leverages deep learning to revolutionize wildfire detection. By utilizing the vast potential of real-time satellite data, the system is designed to promptly identify wildfire-affected regions, offering a proactive approach to mitigation. This report encapsulates the journey of developing a robust model for early wildfire detection and outlines a method for estimating $CO_2$ emissions and burned areas using satellite imagery, contributing to the realms of environmental conservation and disaster management.

In the past, the methods employed for wildfire detection were inherently limited by factors such as monitoring delays and the restricted geographic coverage of ground-based observation systems. Yet, the dawn of modern technology and innovation has unlocked a realm of more effective solutions. These solutions promise to redefine the way we confront wildfires and mitigate their effects.

This project is a testament to this transformative potential. It harnesses the formidable power of satellite imagery, an invaluable tool in contemporary environmental monitoring, and couples it with the remarkable capabilities of deep learning. Together, they promise to revolutionize the field of wildfire detection, offering a beacon of hope in the face of this relentless global threat. By seamlessly integrating the vast potential of real-time satellite data, this system has been meticulously designed to recognize, in a timely manner, regions afflicted by wildfires. It is this proactive approach to early detection that holds the promise of mitigating the devastating impact of wildfires, safeguarding the environment, and protecting human lives.

This report chronicles the incredible journey of developing a robust model for early wildfire detection. It outlines a sophisticated methodology for the estimation of critical parameters such as CO2 emissions and the extent of burned areas through the analysis of satellite imagery. By doing so, it brings to light the pivotal role that technology can play in the realms of environmental conservation and disaster management.

As the global community confronts the urgent need for innovative solutions to the wildfire crisis, this project not only offers a groundbreaking approach but also stands as a testament to human ingenuity and determination. It is a testament to our capacity to adapt, innovate, and work toward a more secure and sustainable future for our planet and all its inhabitants.

# FLOWCHART

## Data Flow Diagram

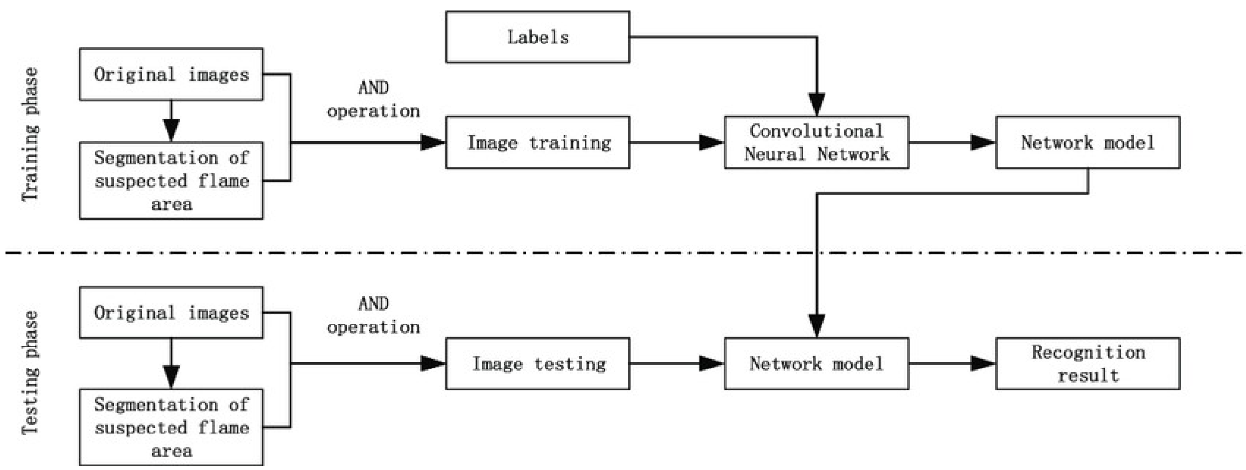


Fig1.1. Data Flow Diagram

# METHODOLOGY

## 2.1 Software Requirements:

Python 3 -We have used Python which is a statistical mathematical programming language like R instead of MATLAB due to the following reasons:

1. Python code is more compact and readable than MATLAB
2. The python data structure is superior to MATLAB
3. It is an open source and also provides more graphic packages and data sets

GitHub and Stack overflow was used for reference in case of programming syntax errors.

## 2.2 Hardware Requirements:

Processor: Intel® Core™ i5-2350M CPU @ 2.30GHz

Installed memory (RAM):8.00GB

System Type: 64-bit Operating System

## 2.3 Kaggle dataset:

Source: https://archive.ics.uci.edu/ml/datasets/forest+fires

Created by: Paulo Cortez and Anbal Morais (Univ. Minho) @ 2007

## 2.4 Training and Testing Data

The methodology involves using satellite imagery and a deep learning model for wildfire detection. The key steps include data preprocessing, model training, prediction, and user interaction through the application.

### 1. Unet Architecture

In this system users' interest and profile features are taken into considerations. They assume that if a one has interests in an item, then one will once again be interested in it in the future. Items sharing something in common are usually grouped together based on their features. User profiles are constructed using past interactions or by indirectly asking users about their interests. Other systems, which are not content-based, utilize user personal and social data. One major issue with this recommendation system is that it makes obvious recommendations because of excessive specialization i.e user A is merely curious about categories B, C, and D, and therefore the system is not ready to recommend movies outside those categories, even though they might be interesting to them. Also, as new users get enrolled, they do not have a proper profile unless they are explicitly asked for information. Nevertheless, it is relatively simple to feature new movies to the system. We just need to ensure that we assign them a group according to their features.

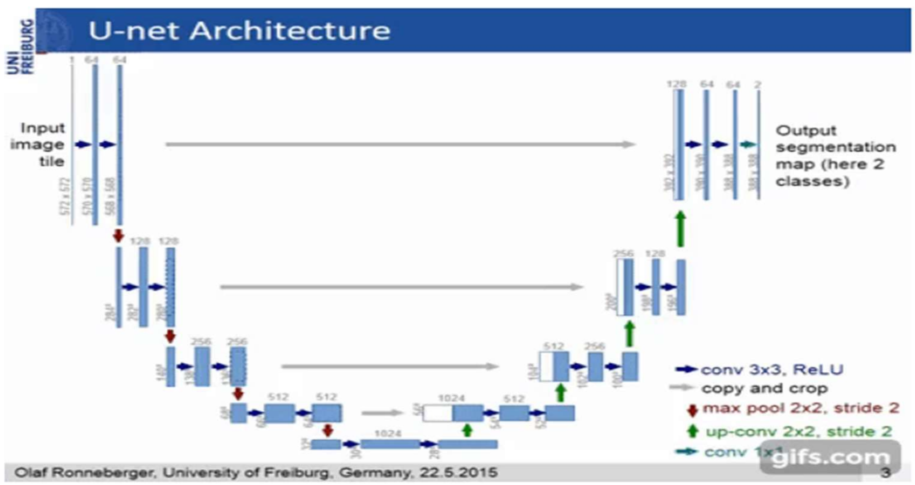


Fig 2.1 Unit Architecture:

**2. Data Collection**

Satellite Imagery: The project relies on satellite imagery as the primary data source. This imagery is collected in real-time from various Earth observation satellites. The dataset includes both raw satellite images and associated labels for areas affected by wildfires.

**3. Data Preprocessing**

Dataset Preprocessing: Before the data can be used for model training and predictions, it undergoes several preprocessing steps:

Batch Image Download: The project uses the Google Image API to batch download satellite images. These images are obtained from multiple sources and cover a wide geographical area.

Labeling Burning Scar Zones: The dataset is enriched by manually contouring and labeling the areas affected by wildfires. This ground truth data is essential for training the deep learning model.

Image Resizing and Padding: To ensure uniformity in the dataset, the images are resized to a consistent format. Padding is applied as needed to make all images compatible for processing.

Data Augmentation: Data augmentation techniques such as rotation, flipping, and brightness adjustments may be used to increase the diversity of the training dataset.

**4. Deep Learning Model: U-Net**

U-Net Architecture: The core of the system is the U-Net deep learning model, which is well-suited for image segmentation tasks. It's a convolutional neural network that learns to identify and delineate objects within images.

Training: The U-Net model is trained using a portion of the dataset that includes the satellite images and their corresponding labeled burning scar zones. During training, the model learns to identify patterns and features associated with wildfires.

**5. Prediction Results**

Making Predictions: Once the U-Net model is trained, it can be used to make predictions on new, unseen satellite images. These predictions are based on the patterns and features learned during training.

Probability Image: The system generates a predicted probability image that highlights areas where wildfires are likely to be occurring. This image provides a visual representation of the model's confidence in its predictions.

Binary Mask: A binary mask is generated from the predictions, indicating the precise boundaries of the burning area. This binary mask can be used for further analysis and assessment.

**6. Use the Application**

Deployment with Streamlit: The system is deployed using Streamlit, a web application framework. Users can interact with the application through a user-friendly interface.

User Interaction: Users can upload their own raw satellite images via the application. These images can be from any location and time. The application allows users to directly drag and drop the image for analysis.

CO2 Emission Calculator: In addition to wildfire detection, the system includes a CO2 emission calculator. Users are prompted to select the type of forest (e.g., Tropical, Temperate, Boreal), and they can input the resolution value. Based on these parameters and the binary mask of the burning area, the system estimates the total burnt area and the CO2 emissions associated with the wildfire.

**7. Presentation of Results**

Display of Predictions: The application presents the prediction results to the user, including the predicted probability image and the binary mask. Users can visualize the areas of interest and assess the model's accuracy.

CO2 Emission Information: The application also provides information about the estimated CO2 emissions and their equivalent in terms of days of electricity power emissions in California.

# IMPLEMENTATION AND RESULT

## Input/Output
## Code:

**Source.py**

```
import numpy as np

from tensorflow.keras.models import load_model

from keras import backend as K

from PIL import Image

def add_image_magin(input_image, im_width=720, im_height=480, color=0):

  # padd

  height, width, vec = input_image.shape

  new_image = np.pad(input_image, ((0, im_height-height), (0, im_width-width), (0, 0)), 'constant',
constant_values=color)

  return new_image

# segment the image

def preprocess_input_image(input_image, im_height=480, im_width=720):

  # normalize the pixel value

  input_image = input_image / np.max(input_image.astype('float'))

  height, width, vec = input_image.shape

  row_num = height//im_height if height%im_height==0 else height//im_height+1

  col_num = width//im_width if width%im_width==0 else width//im_width+1

  new_image_array = np.zeros((row_num*col_num, im_height, im_width, vec))


  for i in range(row_num):

    for j in range(col_num):
```

```
temp_image =
input_image[im_height*i:im_height*(i+1), im_width*j:im_width*(j+1), :]

new_image_array[i*col_num+j] = add_image_magin(temp_image, im_width=im_width, im_height=im_height,
color=0)

  return new_image_array, row_num, col_num

# combine all images back to original size

def combine_image(input_image, row_num, col_num, original_height, original_width, im_height=480, im_width=720,
remove_ghost=True):

  num, height, width, vec = input_image.shape

  new_image = np.zeros((height*row_num, width*col_num, vec))

  for i in range(row_num):

    for j in range(col_num):

      # Remove the ghost caused by CNN model prediction boundary artifacts.

      # by padding same 4 pixels on the boundary

      if remove_ghost:

        # padd all four edges

        input_image[i*col_num+j, :, :, :] = np.pad(input_image[i*col_num+j, 4:height-4, 4:width-4, :], ((4, 4), (4, 4), (0,
0)), 'edge')


      new_image[im_height*i:im_height*(i+1), im_width*j:im_width*(j+1), :] = input_image[i*col_num+j, :, :, :]

  return new_image[:original_height, :original_width, :]


# predict batches of images, and return the probability array

def batch_predict(input_image_array, model):

  num, height, width, vec = input_image_array.shape

  preds_array = np.zeros((num, height, width, 1))

  # to avoid OOM
```

```
  for ii in range(input_image_array.shape[0]):
    preds_array[ii] = model.predict(np.expand_dims(input_image_array[ii, :, :, :], axis=0), verbose=1)
  return preds_array


def conv_float_int(image):
  # convert arry back to [0, 255] for display
  if not np.any(image): # check if the array is all zero
    return image.astype(int)
  return ((image-np.min(image))/(np.max(image)-np.min(image))*255).astype(int)


def load_trained_model(model_location):
    loaded_model = load_model(model_location)
    session = K.get_session()
    return loaded_model, session


def burn_area(output_mask, resolution, forest_type):
  # reference: https://www.geosci-model-dev.net/4/625/2011/gmd-4-625-2011.pdf
  # unit in g/m^2
  biomass_type = {'Tropical Forest': 28076,
                 'Temperate Forest':10492,
                 'Boreal Forest': 25000,
                 'Shrublands': 5705,
                 'Grasslands': 976
                }
  area = np.count_nonzero(output_mask) * resolution**2
```

```
# Ei = A(x,t)×B(x)×FB×efi,
# A: area,
# B(biomass_type),
# FB:Burning fraction, assume 1 here
# efi: mass of biomass burnt, for CO2, it averages 1624 g/kg.
biomass_burnt =  area * biomass_type[forest_type]/1e3 * 1624 #unit in g


# Califorlia annual CO2 emision from power generating, 424 million metric tons of CO2 per year
# Reference: https://ww3.arb.ca.gov/cc/inventory/pubs/reports/2000_2016/ghg_inventory_trends_00-16.pdf
ca_co2_daily = 4.24e8 / 365.


equal_days = biomass_burnt /1e6 / ca_co2_daily


print('The total burnt area is:', "{:.4e}".format(area/1e6), 'km^2 \n')
print('The total CO2 emitted is:', "{:.4e}".format(biomass_burnt/1e6), 'tons \n')
print("Which equivalent to:", "{:.4e}".format(equal_days), " days of Califorlia's  daily electricity power emission \n")
return area, biomass_burnt, equal_days
```

**App.py**

```
import streamlit as st
import tensorflow as tf
import matplotlib.pyplot as plt
from PIL import Image
from source import preprocess_input_image, batch_predict, conv_float_int, combine_image, load_trained_model,
burn_area
import numpy as np
from keras import backend as K
from tensorflow.python.lib.io import file_io
from keras.models import load_model
with open("style.css") as f:
    st.markdown('<style>{}</style>'.format(f.read()), unsafe_allow_html=True)
st.title(" Wild Fire Detection ")
st.sidebar.markdown("** App Status **")
model, session = load_trained_model("temp_model.h5")
K.set_session(session)


st.sidebar.markdown('Please upload a raw satellite image')
uploaded_file = st.sidebar.file_uploader("Upload png file", type=["png"])
if uploaded_file is not None:
   uploaded_image = Image.open(uploaded_file)
    st.markdown("** Original Raw Image: **")
    st.image(uploaded_image, width = 500)


    ### Preprocess the raw image
```

```
#st.sidebar.text("Pre-processing the image...")
    with st.spinner("Pre-processing the image..."):
        input_image_array = np.array(uploaded_image)
        original_width, original_height, pix_num = input_image_array.shape
        new_image_array, row_num, col_num = preprocess_input_image(input_image_array)
        st.sidebar.success("Pre-processing has been done.")
    with st.spinner("Making the prediction..."):
        #### Make Prediction
        preds = batch_predict(new_image_array, model)
        # combine the images, and converted to 0-255 for display
        output_pred = conv_float_int(combine_image(preds, row_num, col_num, original_width, original_height, remove_ghost=True)[:,:,0])
        st.sidebar.success("Prediction has been done.")
        # add image mask to the probability array

    #### Show the picture
    st.markdown("** The Predicted Probability is **:" )
    plt.imshow(output_pred)
    #st.pyplot(output_pred)
    #threshold = st.sidebar.slider("Threshold", 0, 1, 0.25)
   preds_t = (preds > 0.25).astype(np.uint8)
    output_mask = conv_float_int(combine_image(preds_t, row_num, col_num, original_width, original_height, remove_ghost=False)[:,:,0])
    st.markdown("** The Predicted Mask is **: ")
    plt.imshow(output_mask)
    #st.pyplot(output_mask)
    st.image(output_mask)
```

```
#plt.imshow(output_mask)


    st.sidebar.markdown("** CO2 Emission Calculator **")

    forest_type = st.sidebar.selectbox("Please select the type of forest: ", ('Tropical Forest', 'Temperate Forest', 'Boreal Forest', 'Shrublands', 'Grasslands'))

    resolution = st.sidebar.text_input("Please enter the image resolution value: ", '10')


    area, biomass_burnt, equal_days = burn_area(output_mask = output_mask, resolution = float(resolution), forest_type = forest_type)

    st.sidebar.markdown('The total burnt area is:')

    st.sidebar.text("{0:.2f}".format(area/1e6) + ' km^2')

    st.sidebar.markdown('The total CO2 emitted is:')

    st.sidebar.text("{0:.2f}".format(biomass_burnt/1e6) + ' tons')

    st.sidebar.markdown("This is equivalent to:")

    st.sidebar.text("{0:.2f}".format(equal_days) + " days of Califorlia's daily electricity power emission")
```

**Output:**

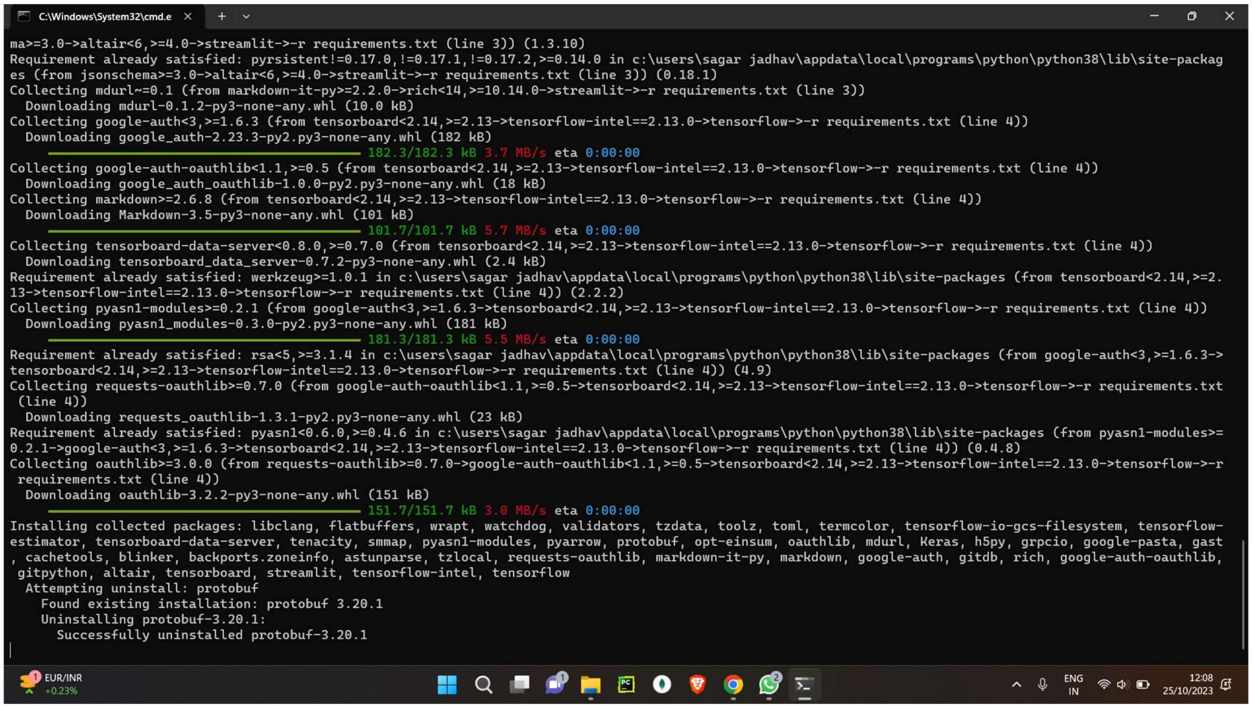


*Fig 1: Initialization of Streamlit and other necessary packages.*

Here we have installed required packages like Streqamlit,google-auth,outhlib etc.These packages are used for this project. We have imported these libraries. We also have ensured package dependencies.

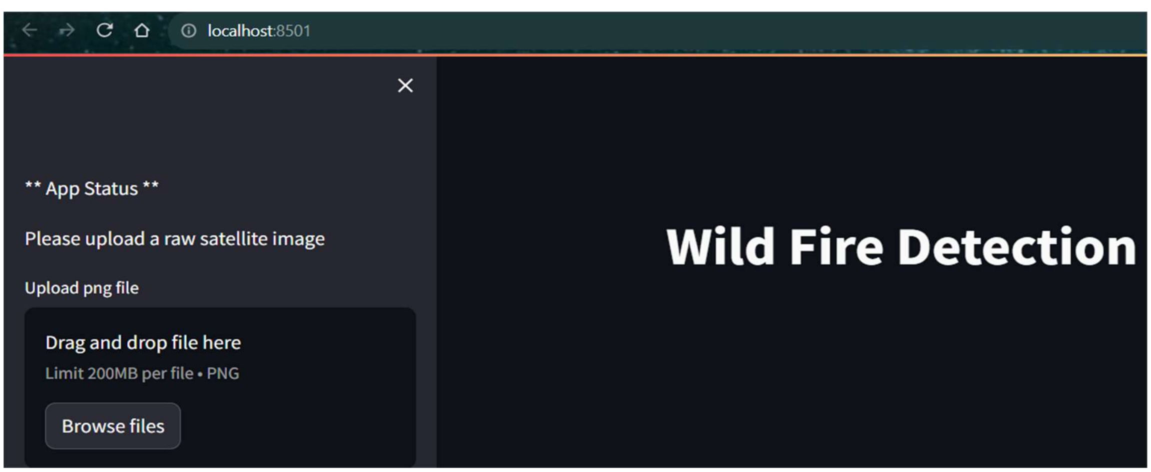


*Fig 2:Dashboard where satellite images are uploaded and then processed under deep learning.*

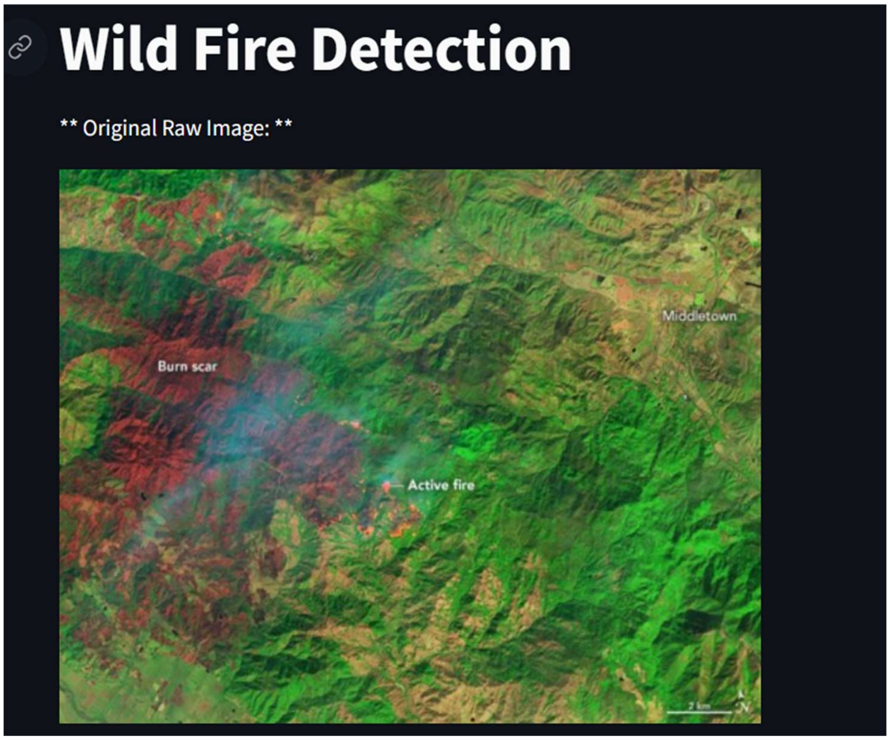


*Fig 3:Uploaded image is being displayed on the dashboard.*

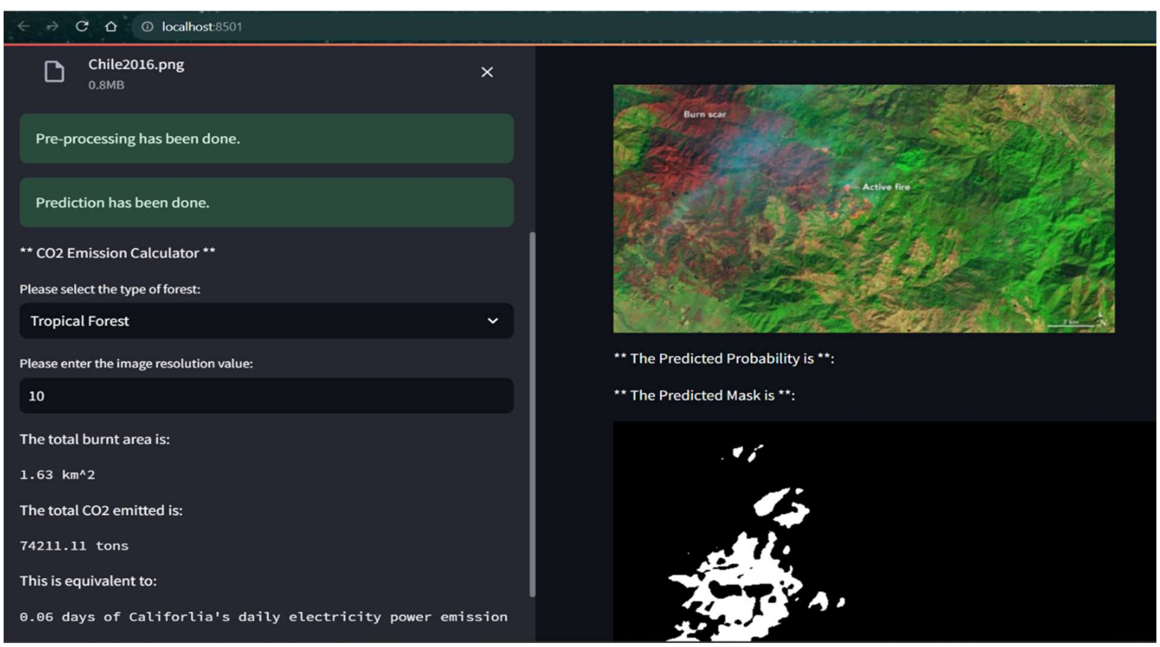


*Fig 4: Here we enter the type of image and other factors.*

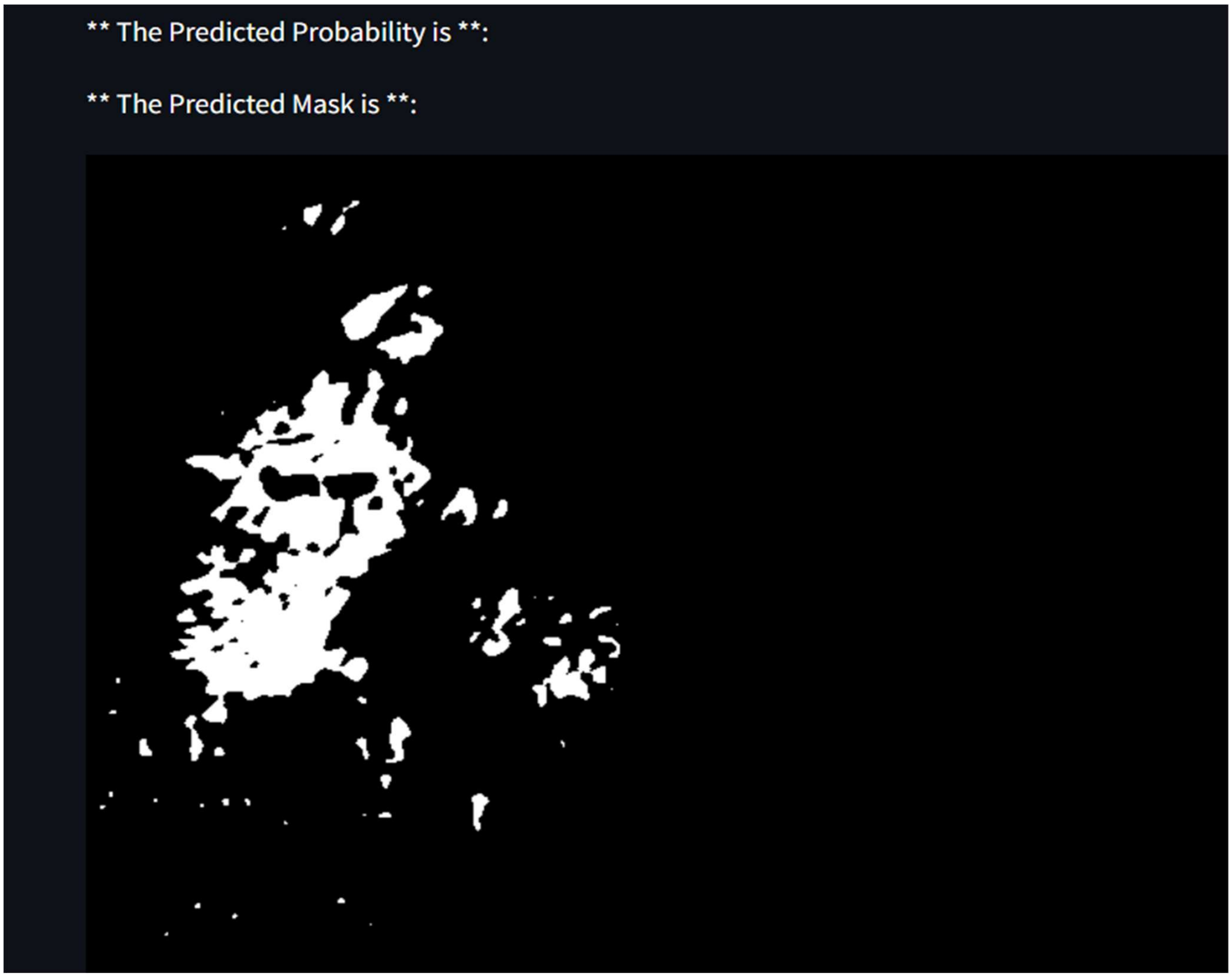


*Fig 5: Predicted result of the original image.*

Here, we can see the burnt area (White color) from the uploaded image.

## CONCLUSION

The Wildfire Real-Time Detection System presented in this project offers an efficient and accurate solution for early wildfire detection using satellite imagery. By integrating deep learning techniques, it can provide real-time information on wildfire-affected areas, enabling timely response and assessment of the environmental impact. This system contributes to better wildfire management and environmental conservation.

By accurately identifying wildfire-affected regions and estimating their environmental impact, this project contributes to the fields of environmental conservation and disaster management. The system's real-time capabilities make it a valuable tool for minimizing the catastrophic consequences of wildfires and protecting both the environment and human well-being.

## REFERENCES

[1][Abid, F. 2021. A survey of machine learning algorithms based forest fires prediction and detection systems. Fire Technology 57 (2): 559–590.

[2]Anh, N. D., P. Van Thanh, D. T. Lap, N. T. Khai, T. Van An, T. D. Tan, N. H. An, and D. N. Dinh. 2022. Efficient forest fire detection using rule-based multi-color space and correlation coefficient for application in unmanned aerial vehicles. *KSII Transactions on Internet and Information Systems (TIIS)* 16 (2): 381–404.

[3]Arif, M., K. Alghamdi, S. Sahel, S. Alosaimi, M. Alsahaft, M. Alharthi, and M. Arif. 2021. Role of machine learning algorithms in forest fire management: a literature review. *J Robotics Autom* 5 (1): 212–226.